JOBSHEET 11
LINKED LIST

Mata Kuliah : Algoritma dan Struktur Data

Dosen : **Mungki Astiningrum, S.T., M.Kom.**

ILHAM DHARMA ATMAJA
TI 1A
(244107020220)

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG TAHUN 2025

1. Single Linked List
    1) Clas Mahasiswa14

```java
public class Mahasiswa14 {

    String nim, nama, kelas;

    double ipk;


    public Mahasiswa14() {

    }


    public Mahasiswa14(String nim, String nama, String kelas, double ipk)
{

        this.nim = nim;

        this.nama = nama;

        this.kelas = kelas;

        this.ipk = ipk;

    }


    public void tampilInformasi() {

        System.out.println("NIM   : " + nim);

        System.out.println("Nama  : " + nama);

        System.out.println("Kelas : " + kelas);

        System.out.println("IPK   : " + ipk);

    }

}
```

    2) Class NodeMahasiswa14

```java
public class NodeMahasiswa14 {

    Mahasiswa14 data;

    NodeMahasiswa14 next;


    public NodeMahasiswa14(Mahasiswa14 data, NodeMahasiswa14 berikutnya) {

        this.data = data;

        this.next = berikutnya;

    }

}
```

3) Class SingleLinkedList14

```java
public class SingleLinkedList14 {
    NodeMahasiswa14 head, tail;

    public SingleLinkedList14() {
        head = null;
        tail = null;
    }

    public boolean isEmpty() {
        return head == null;
    }

    public void print() {
        if (!isEmpty()) {
            NodeMahasiswa14 tmp = head;
            System.out.println("Isi Linked List:");
            while (tmp != null) {
                tmp.data.tampilInformasi();
                tmp = tmp.next;
            }
            System.out.println("");
        } else {
            System.out.println("Linked List kosong");
        }
    }
    public void addFirst(Mahasiswa14 input) {
        NodeMahasiswa14 ndInput = new NodeMahasiswa14(input, null);
        if (isEmpty()) {
            head = ndInput;
            tail = ndInput;
        } else {
            ndInput.next = head;
            head = ndInput;
        }
    }
```

```java
        public void addLast(Mahasiswa14 input) {
        NodeMahasiswa14 ndInput = new NodeMahasiswa14(input, null);
        if (isEmpty()) {
            head = ndInput;
            tail = ndInput;
        } else {
            tail.next = ndInput;
            tail = ndInput;
        }
    }
    public void insertAfter(String key, Mahasiswa14 input) {
        NodeMahasiswa14 temp = head;
        do {
            if (temp.data.nim.equals(key)) {
                NodeMahasiswa14 ndInput = new NodeMahasiswa14(input, temp.next);
                temp.next = ndInput;
                if (ndInput.next == null) tail = ndInput;
                break;
            }
            temp = temp.next;
        } while (temp != null);
    }

    public void insertAt(int index, Mahasiswa14 input) {
        if (index < 0) {
            System.out.println("Index tidak valid!");
        } else if (index == 0) {
            addFirst(input);
        } else {
            NodeMahasiswa14 temp = head;
            for (int i = 0; i < index - 1; i++) {
                if (temp == null) break;
                temp = temp.next;
            }
            if (temp != null) {
                NodeMahasiswa14 ndInput = new NodeMahasiswa14(input, temp.next);
                temp.next = ndInput;
                if (ndInput.next == null) tail = ndInput;
            } else {
                System.out.println("Index melebihi ukuran list!");
            }
        }
    }
}
```

4) Class SLLMain14

```
public class SLLMain14 {
    public static void main(String[] args) {
        SingleLinkedList14 sll = new SingleLinkedList14();

        Mahasiswa14 mhs1 = new Mahasiswa14("2341010014", "Dewi", "2A", 3.5);
        Mahasiswa14 mhs2 = new Mahasiswa14("2341010015", "Raka", "2B", 3.8);
        Mahasiswa14 mhs3 = new Mahasiswa14("2341010016", "Sari", "2A", 3.4);
        Mahasiswa14 mhs4 = new Mahasiswa14("2341010017", "Dirga", "2C", 3.6);

        sll.print();
        System.out.println("");

        sll.addFirst(mhs4);
        sll.print();
        System.out.println("");

        sll.addLast(mhs1);
        sll.print();
        System.out.println("");

        sll.insertAfter("2341010017", mhs3);
        sll.insertAt(2, mhs2);
        sll.print();
    }
}
```

1.2 Pertanyaan

1. Karena pada saat baris tersebut dieksekusi, belum ada data/node yang dimasukkan ke dalam linked list.
2. Variabel temp digunakan sebagai penunjuk (pointer) **sementara** untuk menelusuri (traverse) elemen-elemen dalam linked list.
   Biasanya digunakan untuk: mengecek isi list dari awal hingga akhir dan
   Mencari node tertentu
3.

```
import java.util.Scanner;


public class SLLMain14 {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        SingleLinkedList14 sll = new SingleLinkedList14();


        System.out.print("Masukkan jumlah data mahasiswa: ");

        int jumlah = sc.nextInt();

        sc.nextLine(); // consume newline


        for (int i = 0; i < jumlah; i++) {

            System.out.println("\nData Mahasiswa ke-" + (i + 1));

            System.out.print("NIM: ");

            String nim = sc.nextLine();

            System.out.print("Nama: ");

            String nama = sc.nextLine();

            System.out.print("Kelas: ");

            String kelas = sc.nextLine();

            System.out.print("IPK: ");

            double ipk = sc.nextDouble();

            sc.nextLine(); // consume newline


            Mahasiswa14 mhs = new Mahasiswa14(nim, nama, kelas, ipk);

            sll.addLast(mhs);

            sll.print();

        }

    }

}
```

2. Modifikasi Elemen pada Single Linked List
    1) Class SingleLinkedList14

```
public Mahasiswa14 getData(int index) {
    NodeMahasiswa14 temp = head;
    for (int i = 0; i < index; i++) {
        if (temp == null) return null;
        temp = temp.next;
    }
    return temp != null ? temp.data : null;
}

public int indexOf(String key) {
    NodeMahasiswa14 temp = head;
    int index = 0;
    while (temp != null) {
        if (temp.data.nim.equals(key)) {
            return index;
        }
        temp = temp.next;
        index++;
    }
    return -1;
}
public void removeFirst() {
    if (isEmpty()) {
        System.out.println("Linked List masih kosong, tidak dapat dihapus");
    } else if (head == tail) {
        head = tail = null;
    } else {
        head = head.next;
    }
}

public void removeLast() {
    if (isEmpty()) {
        System.out.println("Linked List masih kosong, tidak dapat dihapus");
   } else if (head == tail) {
        head = tail = null;
    } else {
        NodeMahasiswa14 temp = head;
        while (temp.next != tail) {
            temp = temp.next;
        }
        temp.next = null;
        tail = temp;
    }
}
public void remove(String key) {
    if (isEmpty()) {
        System.out.println("Linked List kosong");
        return;
    }

    if (head.data.nim.equals(key)) {
        removeFirst();
        return;
    }

    NodeMahasiswa14 prev = head;
    NodeMahasiswa14 curr = head.next;

    while (curr != null) {
        if (curr.data.nim.equals(key)) {
            prev.next = curr.next;
            if (curr == tail) {
                tail = prev;
            }
            break;
        }
        prev = curr;
        curr = curr.next;
    }
}
```

```
        public void removeAt(int index) {
         if (index == 0) {
             removeFirst();
         } else {
             NodeMahasiswa14 temp = head;
             for (int i = 0; i < index - 1; i++) {
                 if (temp == null || temp.next == null) return;
                 temp = temp.next;
             }
             if (temp.next != null) {
                 temp.next = temp.next.next;
                 if (temp.next == null) {
                     tail = temp;
                 }
             }
         }
     }
}
```

2.2 . Pertanyaan

1. Keyword break digunakan untuk menghentikan proses perulangan (while atau for) saat node yang ingin dihapus sudah ditemukan.
   Tujuannya agar:
   - Tidak perlu terus menelusuri seluruh linked list setelah data ditemukan.
   - Meningkatkan efisiensi program.
2. Kode tersebut digunakan untuk memperbarui pointer tail ketika node yang dihapus adalah node terakhir (paling belakang).

3. Tugas

1) Class QueueLinkedList14

```
public class QueueLinkedList14 {

    NodeMahasiswa14 front, rear;

    int size;


    public QueueLinkedList14() {

        front = rear = null;

        size = 0;

    }


    public boolean isEmpty() {

        return front == null;

    }


    public void enqueue(Mahasiswa14 data) {

        NodeMahasiswa14 newNode = new NodeMahasiswa14(data, null);

        if (isEmpty()) {

            front = rear = newNode;

        } else {

            rear.next = newNode;

            rear = newNode;

        }

        size++;

        System.out.println("Antrian ditambahkan.");

    }


    public void dequeue() {

        if (isEmpty()) {

            System.out.println("Antrian kosong, tidak bisa memanggil.");

        } else {

            System.out.println("Memanggil:");

            front.data.tampilInformasi();

            front = front.next;

            if (front == null) {

                rear = null;
```

```java
            }
            size--;
        }
    }

    public void peekFront() {
        if (isEmpty()) {
            System.out.println("Antrian kosong");
        } else {
            System.out.println("Antrian terdepan:");
            front.data.tampilInformasi();
        }
    }

    public void peekRear() {
        if (isEmpty()) {
            System.out.println("Antrian kosong");
        } else {
            System.out.println("Antrian terakhir:");
            rear.data.tampilInformasi();
        }
    }

    public void printQueue() {
        if (isEmpty()) {
            System.out.println("Antrian kosong");
        } else {
            System.out.println("Isi antrian:");
            NodeMahasiswa14 temp = front;
            while (temp != null) {
                temp.data.tampilInformasi();
                System.out.println("-------------------");
                temp = temp.next;
            }
        }
    }
    public void clearQueue() {
        front = rear = null;
        size = 0;
        System.out.println("Antrian dikosongkan.");
    }

    public int getSize() {
        return size;
    }
}
```

2) Class

```
import java.util.Scanner;

public class QueueMain14 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        QueueLinkedList14 queue = new QueueLinkedList14();
        int pilih;

        do {
            System.out.println("\n===== MENU ANTRIAN KEMAHASISWAAN =====");
            System.out.println("1. Tambah Antrian");
            System.out.println("2. Panggil Antrian");
            System.out.println("3. Lihat Antrian Terdepan");
            System.out.println("4. Lihat Antrian Terakhir");
            System.out.println("5. Tampilkan Semua Antrian");
            System.out.println("6. Lihat Jumlah Antrian");
            System.out.println("7. Kosongkan Antrian");
            System.out.println("0. Keluar");
            System.out.print("Pilih menu: ");
            pilih = sc.nextInt(); sc.nextLine();

            switch (pilih) {
                case 1:
                    System.out.println("\nInput Data Mahasiswa:");
                    System.out.print("NIM   : ");
                    String nim = sc.nextLine();
                    System.out.print("Nama  : ");
                    String nama = sc.nextLine();
                    System.out.print("Kelas : ");
                    String kelas = sc.nextLine();
                    System.out.print("IPK   : ");
                    double ipk = sc.nextDouble(); sc.nextLine();
                    Mahasiswa14 mhs = new Mahasiswa14(nim, nama, kelas, ipk);
                    queue.enqueue(mhs);
                    break;
                case 2:
                    queue.dequeue();
                    break;
                case 3:
                    queue.peekFront();
                    break;
                case 4:
                    queue.peekRear();
                    break;
                case 5:
                    queue.printQueue();
                    break;
                case 6:
                    System.out.println("Jumlah antrian: " + queue.getSize());
                    break;
                case 7:
                    queue.clearQueue();
                    break;
                case 0:
                    System.out.println("Terima kasih!");
                    break;
                default:
                    System.out.println("Pilihan tidak valid!");
            }
        } while (pilih != 0);
    }
}
```

Link Git Hub

https://github.com/ianmen10/SEMESTER-genap2.git